IN THE CLAIMS:

Please re-write the claims to read as follows:

Please cancel claims 1-66 without prejudice.

1-66. (Canceled).

1

Please enter the following new claims 67-71 et seq.

- 1 67. (New): A processor, comprising:
- a first execution unit having a first and second input register coupled to first and
- second inputs to a first arithmetic logic unit (ALU), the first and second input registers of
- 4 the first execution unit to store source operands;
- a second execution unit having a first and second input register, the second regis-
- 6 ter coupled to a second input to a second ALU, the first and second input registers of the
- 7 second execution unit to store source operands; and
- a mulitplexer (MUX) having i) a first input coupled to the first input of the first
- 9 ALU, ii) a second input coupled to the first input register of the second ALU, and iii) an
- output directly providing a first input to the second ALU, the MUX permitting both the
- first and second ALU to share the source operand stored in the first input register of the
- 12 first ALU.
- 1 68. (New): The processor of claim 67, further comprising:
- an instruction set defining a register decode value that specifies source operand
- bypassing, such that the MUX, in response to the register decode value that specifies

- source operand bypassing, selects the first input of the MUX coupled to the first input of
- 5 the first ALU as the output of the MUX, the output of the MUX providing the first input
- 6 to the second ALU.
- 69. (New): The processor of claim 68, wherein the source operand bypassing value al-
- lows the second execution unit to receive data stored at an effective memory address
- specified by a displacement operand in the previous instruction executed by the first exe-
- 4 cution unit.
- 1 70. (New): The processor of claim 67, further comprising:
- a local bus for communicating with a memory;
- a register file for storing intermediate operands; and
- an instruction decode stage for coupling the register file to the first and second in-
- 5 put registers of the first and second ALUs to provide intermediate operands as the source
- operands, and for coupling a memory bus to the first input register of the of the first ALU
- 7 to provide source operands from the memory.
- 71. (New): The processor of claim 70, wherein the first input register of the first ALU
- 2 provides source operands from the memory to both the first input to the first ALU and to
- the first input of the MUX, thereby permitting the first input to the second ALU to share
- 4 the source operands from the memory directly from the first input register of the first
- 5 ALU.
- 72. (New): The processor of claim 67, further comprising:

- a pipeline of the processor, the pipeline having a plurality of stages including in-
- struction decode, writeback, and execution stages, the execution stage having the first and
- 4 second execution units.
- 1 73. (New): The processor of claim 72, further comprising:
- an instruction set defining a register decode value that defines result bypassing
- that allows bypassing of a result from a previous instruction executing in pipeline stages
- of the processor by directly addressing a result register of the first and second execution
- 5 units.
- 74. (New): The processor of claim 73, wherein the register decode value comprises:
- one of a result bypass (RRB) operand and an inter-unit result bypass (RIRB) op-
- erand, each of which explicitly controls data flow within the pipeline of the processor.
- 75. (New): The processor of claim 74, wherein the RRB operand denotes the first exe-
- 2 cution unit and the RIRB operand denotes the second execution unit.
- 76. (New): The processor of claim 74, wherein the RRB operand explicitly infers feed-
- 2 back of the data delivered from the first execution unit to an input register of the first
- 3 execution unit over a feedback path.
- 77. (New): The processor of claim 76, wherein the writeback stage comprises an inter-
- stage register and wherein the RRB operand enables bypassing write-back of the data
- processed by the first and second execution units to one of the register file or the inter-
- 4 stage register of the writeback stage.

- 1 78. (New): The processor of claim 67, further comprising:
- an instruction set defining a register decode value that defines source operand by-
- passing that allows source operand data to be shared among the first and second execu-
- tion units by directly addressing a source register of the first execution unit.
- 79. (New): The processor of claim 78, wherein the source operand bypassing value al-
- lows the second execution unit to receive data stored at an effective memory address
- specified by a displacement operand in the previous instruction executed by the first exe-
- 4 cution unit.
- 1 80. (New): The processor of claim 67, further comprising:
- a register file containing a plurality of general-purpose registers for storing inter-
- mediate result data processed by the first and second execution units.
- 1 81. (New): The processor of claim 67, wherein the first and second execution units are
- 2 parallel execution units.
- 82. (New): The processor of claim 67, further comprising:
- a current execution unit as the first execution unit; and
- an alternate execution unit as the second execution unit.
- 83. (New): The processor of claim 67, wherein the first and second ALU share the
- source operand stored in the first input register of the first ALU substantially simultane-
- 3 ously.

- 1 84. (New): A processor, comprising:
- a first arithmetic logic unit (ALU);
- a second ALU;
- a mulitplexer (MUX) having i) a first input coupled to a first input of the first
- 5 ALU, ii) a second input coupled to source operands, and iii) an output providing a first
- 6 input to the second ALU, the MUX permitting both the first and second ALU to share the
- 7 same source operand; and
- an instruction set defining an instruction that when decoded operates the MUX to
- 9 permit both first and second ALUs to share the same source operand.
- 1 85. (New): The processor of claim 84, further comprising:
- a local bus for communicating with a memory, wherein the first and second ALUs
- share the same source operand from memory.
- 86. (New): The processor of claim 84, further comprising:
- a register decode value of the instruction set that defines source operand bypass-
- 3 ing that allows source operand data to be shared among the first and second ALUs by di-
- 4 rectly addressing a source register of the first ALU.
- 87. (New): The processor of claim 86, wherein the source operand bypassing value al-
- lows the second ALU to receive data stored at an effective memory address specified by
- a displacement operand in the previous instruction executed by the first ALU.
- 88. (New): The processor of claim 84, further comprising:

- a register file containing a plurality of general-purpose registers for storing inter-
- mediate result data processed by the first and second ALUs.
- 89. (New): The processor of claim 84, wherein the first and second ALUs are parallel
- 2 ALUs.
- 90. (New): The processor of claim 84, further comprising:
- a current ALU as the first ALU; and
- an alternate ALU as the ALU unit.
- 91. (New): The processor of claim 84, wherein the first and second ALUs share the
- same source operand substantially simultaneously.
- 92. (New) A method for use with a processor, the method comprising:
- providing a mulitplexer (MUX) having a first and second MUX input and a MUX
- 3 output;
- 4 coupling the first MUX input to a first input of a first ALU;
- coupling a second MUX input to a first input register of a second ALU; and
- directly providing, by the MUX output, a first input to the second ALU, the MUX
- permitting both the first and second ALU to share a source operand stored in a first input
- 8 register of the first ALU.
- 93. (New): The method of claim 92, further comprising:

- defining a register decode value within an instruction set that specifies source op-
- erand bypassing, such that the MUX, in response to the register decode value that speci-
- fies source operand bypassing, selects the first MUX input coupled to the first input of
- 5 the first ALU as the MUX output, the MUX output providing the first input to the second
- 6 ALU.
- 94. (New): The method of claim 93, wherein the source operand bypassing value allows
- the second execution unit to receive data stored at an effective memory address specified
- by a displacement operand in the previous instruction executed by the first execution unit.
- 1 95. (New): The method of claim 92, further comprising:
- 2 communicating with a memory;
- storing intermediate operands in a register file; and
- 4 identifying an instruction decode stage for coupling the register file to the first
- and second input registers of the first and second ALUs to provide intermediate operands
- as source operands, and for coupling a memory bus to the first input register of the of the
- 7 first ALU to provide source operands from the memory.
- 96. (New): The method of claim 95, further comprising:
- 2 providing, by the first input register of the first ALU, source operands from the
- memory to both the first input to the first ALU and to the first MUX input, thereby per-
- 4 mitting the first input to the second ALU to share the source operands from the memory
- directly from the first input register of the first ALU.
- 97. (New): The method of claim 92, further comprising:

- including a pipeline of the processor, the pipeline having a plurality of stages in-
- cluding instruction decode, writeback, and execution stages, the execution stage having a
- 4 first and second execution unit, each having one of the first and second ALUs, respec-
- 5 tively.
- 98. (New): The method of claim 97, further comprising:

defining a register decode value that defines result bypassing of a result from a previous instruction executing in pipeline stages of the processor.

- 1 99. (New): The method of claim 98, further comprising:
- identifying a pipeline stage register for use as a source operand in an instruction
- containing the register decode value by directly addressing a result register.
- 1 100. (New): The method of claim 99, further comprising:
- explicitly controlling data flow within the pipeline stages of the processor through
- the use of a register result bypass (RRB) operand in the register decode value.
- 1 101. (New): The method of claim 100, wherein the step of explicitly controlling com-
- 2 prises:
- retrieving data from the first execution unit; and
- returning the data to an input of the first and second execution units as specified
- by the RRB operand, thereby bypassing write-back of the data to either a register file or
- 6 memory at the writeback stage.

- 102. (New): The method of claim 101, wherein the step of identifying further com-
- 2 prises:
- explicitly specifying the pipeline stage register to be used as the source operand
- 4 for the instruction.
- 1 103. (New): The method of claim 102, further comprising:
- encoding the RRB operand in fewer bits than a regular register operand.
- 1 104. (New): The method of claim 97, further comprising:
- defining a register decode value that defines source operand bypassing of source
- 3 operand data.
- 1 105. (New): The method of claim 104, further comprising:
- identifying a pipeline stage register for use as a source operand in an instruction
- 3 containing the register decode value by directly addressing a source register.
- 1 106. (New): The method of claim 105, further comprising:
- sharing source operand data among the first and second execution units of the
- pipelined processor through the use of a source bypass (RISB) operand in the register de-
- 4 code value.
- 107. (New): The method of claim 106, wherein the step of sharing further comprises:

- receiving data at the second execution unit, the data stored at a memory address
- specified by a displacement operand in a previous instruction executed by the first exe-
- 4 cution unit of the processor.
- 108. (New): The method of claim 107, wherein the step of sharing further comprises:
- realizing two memory references through the use of a single bus operation over a
- 3 local bus.
- 1 109. (New): The method of claim 108, wherein the step of sharing further comprises:
- encoding the RISB operand with substantially fewer bits than those needed for a
- 3 displacement address.
- 1 110. (New): The method of claim 92, further comprising:
- sharing a source operand stored in a first input register of the first ALU at the first
- and second ALU substantially simultaneously.
- 1 111. (New) An apparatus, comprising:
- means for providing a mulitplexer (MUX) having a first and second MUX input
- and a MUX output;
- means for coupling the first MUX input to a first input for a first ALU;
- means for coupling a second MUX input to a first input register for a second
- 6 ALU; and
- means for directly providing, by the MUX output, a first input to the second ALU,
- the MUX permitting both the first and second ALU to share a source operand stored in a
- 9 first input register of the first ALU.

- 1 112. (New): The apparatus of claim 111, further comprising:
- means for defining a register decode value within an instruction set that specifies
- source operand bypassing, such that the MUX, in response to the register decode value
- that specifies source operand bypassing, selects the first MUX input coupled to the first
- input of the first ALU as the MUX output, the MUX output providing the first input to
- 6 the second ALU.
- 1 113. (New): The apparatus of claim 112, wherein the source operand bypassing value
- allows the second execution unit to receive data stored at an effective memory address
- specified by a displacement operand in the previous instruction executed by the first exe-
- 4 cution unit.
- 1 114. (New): The apparatus of claim 111, further comprising:
- 2 means for communicating with a memory;
- means for storing intermediate operands; and
- 4 means for identifying an instruction decode stage for coupling the register file to
- 5 the first and second input registers of the first and second ALUs to provide intermediate
- operands as source operands, and for coupling a memory bus to the first input register of
- the of the first ALU to provide source operands from the memory.
- 1 115. (New): The apparatus of claim 114, further comprising:
- means for providing, by the first input register of the first ALU, source operands
- from the memory to both the first input to the first ALU and to the first MUX input,

- thereby permitting the first input to the second ALU to share the source operands from
- 5 the memory directly from the first input register of the first ALU.
- 1 116. (New): The apparatus of claim 111, further comprising:
- means for sharing a source operand stored in a first input register of the first ALU
- at the first and second ALU substantially simultaneously.
- 1 117. (New) A computer readable media, comprising: the computer readable media con-
- taining instructions for execution in a processor for the practice of the method of,
- providing a mulitplexer (MUX) having a first and second MUX input and a MUX
- 4 output;
- coupling the first MUX input to a first input of a first ALU;
- 6 coupling a second MUX input to a first input register of a second ALU; and
- directly providing, by the MUX output, a first input to the second ALU, the MUX
- permitting both the first and second ALU to share a source operand stored in a first input
- 9 register of the first ALU.